An equivalence on terms for security protocols

R. Ramanujam and S. P. Suresh

The Institute of Mathematical Sciences
C.I.T. Campus, Chennai 600 113, India.
E-mail: {jam,spsuresh}@imsc.res.in

Abstract. Modelling security protocols easily leads us to consideration of infinite state systems and as a result, the verification of secrecy, a crucial requirement in security, becomes undecidable. To cope with this, various bounds are externally imposed to yield finite systems and verification is done for these systems. In this paper, we suggest a semantic approach, whereby the bounds are obtained by equating terms used in communications. We propose an equivalence relation on terms of finite index which leads to a notion of normal terms such that the secrecy problem becomes decidable for those protocols that use only normal terms. Many interesting protocols studied in the literature seem to respect this relation, suggesting that finite state methods may be applicable for these (*a priori*) infinite state systems.

1 Introduction

Cryptographic protocols operate in environments where communicating agents exchange information over public channels. The possibility of intruders listening to message exchanges and manipulating the system (for example by blocking messages, or forging them) necessitates the use of encrypted communication. Stringent requirements on these protocols relate to ensuring secrecy of information (only the receiver gets the secret and not any intruder), authentication (the secret does originate from the purported sender) and so on. Despite considerable ingenuity on the part of the designers of these protocols, many possible attacks are often discovered later. Interestingly, most of these attacks are independent of the encryption schemes used, but rely on logical design flaws, and use intruders' abilities to manipulate communication patterns between honest agents ([Low96], [CJ97]).

It is in this context that formal verification of security protocols assumes importance ([Me95], [SC00]). The protocols are typically finite (indeed, very short) sequences of communications and the requirements can be stated in simple logics (for instance propositional modal logics). Hence the use of theorem proving techniques to verify the correctness of such protocols, as well as model checking for finding attacks, seems a priori interesting and worthwhile ([Bol97], [MCJ97], [Pau98]).

The verification problem for security protocols can be formulated as follows: given an abstract specification of the protocol as a sequence of communications between agents, is it the case that every run generated by possible multi-sessions between agents, with a hypothetical intruder interleaving arbitrarily many actions, satisfy the given security requirements? There are many requirements but an important (and central) requirement is that of *secrecy*: a secret that is generated by an honest agent should not be leaked to the intruder, who is assumed to have unlimited computational resources and can keep a record of every public system event and utilize it at an arbitrarily later time. However, the intruder cannot generate an honest agent's secret autonomously, nor can it break encryption.

Security protocols are typically specified as a (finite) set of roles (typically with names like challenger, responder and so on). These are abstract patterns of communication which specify what messages are sent when, and how to respond to the receipt of any message. The content of these messages is (usually) not relevant, but the structure is; hence abstract variables suffice to describe the protocol. A system consists of a finite set of agents. In any system run, an agent plays one or more roles, each time instantiating variables appropriately with its secrets. Every honest agent is assumed to follow the protocol.

In interesting situations, the same agent may play several roles simultaneously, often with the same agents; these are referred to as multi-sessions. Moreoever, an agent may play a role many times. Hence the set of system runs is (typically) infinite.

Typically, one all-powerful intruder is assumed, who can copy every communication in the system, can block any message and can pretend to be any agent. It is assumed that the intruder has unlimited computational resources and can keep a record of every public system event and utilize it at an arbitrarily later time. However, the intruder cannot generate an honest agent's secret autonomously, nor can it break encryption.

Even the precise modelling of system states is non-trivial. This has to do with the fact that the intruder needs to be unconstrained, and yet the state of knowledge of the intruder is crucial for verification; much of the literature is devoted to this aspect ([Gou00], [Mo99]). The next difficulty is that when we model the semantics of the system precisely, we get **infinite state systems**. There are many sources of unboundedness in the modelling of security protocols.

The first type of unboundedness relates to the requirement of *freshness*: every time an agent sends out a secret (a nonce), it is a new one — an obvious requirement to avoid the intruder *replaying* old sessions. But this means that when there is no bound on the number of plays of roles by agents, the number of nonces used grows unboundedly as well. Unbounded length of messages may also cause complication. Since the intruder may generate arbitrarily long messages, and agents receiving them may be constrained by the protocol to respond in kind, the system state space may become infinite.

Given that the models are infinite state systems, it is not surprising that the secrecy problem for such protocols is undecidable. [DLMS99] use unbounded generation of nonces to show that the secrecy problem for protocols is undecidable, even when the number of roles, the length of each role and message length are bounded. On the other hand, even if only boundedly many nonces are assumed to be generated, the intruder may get unbounded slave work from honest agents using messages of arbitrary length, leading to undecidability as in [HT96].

How then are we to cope with verification of such systems? The literature consists of many proposals that typically place bounds on the number of plays on any run of the protocol, effectively yielding a finite system. Examples of this approach include [ALV01], [MS01] and [RT01]. There are also approaches which impose syntactic restrictions on the use of the tupling operator. Examples of this include [DEK82] and [ALV01]. [CS02] is a good survey on the various approaches to decidability of security protocol verification, and also on the undecidability results.

An alternative to placing such 'external' bounds is to look for subclasses of protocols in which, either by virtue of the manner in which communication patterns between agents are structured, or by the way in which system behaviour is structured, decidability obtains. The definition of such a subclass is arrived at by a detailed analysis of the undecidability proof; while we cannot hope for an exact characterization, it suffices to come up with a restriction that is strong enough to exclude the "source" of undecidability while yet retaining a large enough class of interesting protocols.

In this context, the work [CC03] uses techniques from tree automata to achieve decidability for the class of protocols in which every agent copies at most one piece of any message it receives into any message it sends. (See also [CCM01].)

Our approach in this paper is motivated by semantic considerations. We propose a semantic criterion which gives decidability in the presence of terms of arbitrary length, but with boundedly many nonces. The central contribution of the paper is the definition of a natural equivalence relation on terms, which leads to the notion of normal terms. We define normal protocols to be those for which all the terms appearing in the protocol description are normal and show that, for the class of normal protocols, the secrecy problem is decidable. This is an interesting subclass of protocols, including most protocols standard in the literature ([CJ97]).

It turns out that without the restriction, the halting problem for two-counter machines may be coded, illustrating how distinguishing such terms lies at the source of undecidability. We show that for the subclass studied, the decidability result extends to other properties than secrecy as well, those which can be stated in a simple modal logic.

In a companion paper ([RS03]), we have proposed a simple syntactic restriction on protocols and show decidability in the presence of unboundedly many nonces, but bounded message length. The condition essentially states that between any two terms that occur in distinct communications, no encrypted subterm of one can be unified with a subterm of the other.

2 Security protocols and their semantics

Fix a finite set of *agents* Ag with a special *intruder* $I \in Ag$. $Ag \backslash \{I\}$ is denoted by Ho. The set of *keys* K is $K_{lt} \cup K_{st}$ where K_{lt}, the set of *long-term keys* is the set $\{k_{AB}, pubk_A, privk_A \mid A, B \in Ag, A \neq B\}$, and K_{st} is a finite set of *short-term keys*. $pubk_A$ is A's *public key* and $privk_A$ is its *private key*. k_{AB} is the long-term key *shared* by A and B. For every $k \in K$ define $\overline{k} \in K$ as follows: for the shared keys and short-term keys $\overline{k} = k$, whereas $\overline{pubk_A} = privk_A$ and $\overline{privk_A} = pubk_A$. $\overline{k}$ is k's *inverse key*. For $A \in Ag$, $K_A \stackrel{\text{def}}{=} \{pubk_B, k_{AB} \mid B \neq A\} \cup \{privk_A\}$ is the set of keys known to A. Also fix a finite set of *nonces* N. Define the set of *basic terms* T_0 to be $K \cup N \cup Ag$. (Note that every system studied would have such a finite set T_0 associated with it. Here, for ease of presentation, we fix one such system for the discourse.)

Define the set of information terms to be

$$\mathcal{T} \quad ::= \quad m \mid (t, t') \mid \{t\}_k$$

where m ranges over $T_0 \setminus K_{lt}$ and k ranges over K. We define the set of subterms of a term t, $ST(t)$, to be the least set T such that: $t \in T$; if $(t, t') \in T$ then $t \in T$ and $t' \in T$; and if $\{t\}_k \in T$ then $t \in T$. $ST(T) = \bigcup_{t \in T} ST(t)$ for any $T \subseteq \mathcal{T}$. For a set of terms T and a key k we say that k is *referred to* in T if $k \in T$ or $\exists t : \{t\}_k \in T$.

$\Sigma = \{A!B{:}(M)t, A?B{:}t \mid A, B \in Ag, A \neq B, t \in \mathcal{T}, M \subseteq ST(t) \cap T_0\}$ is the set of *actions*. For $a = A!B{:}(M)t$, $term(a) = t$ and $NT(a) = M$. Similarly for $a = A?B{:}t$, $term(a) = t$ and $NT(a) = \emptyset$. For any action a, $|a|$ is defined to be $|term(a)|$. For any send action $A!B{:}(M)t$, $B?A{:}t$ is said to be its *matching receive*. $terms(a_1 \cdots a_\ell) = \{term(a_i) \mid 1 \leq i \leq \ell\}$ and $NT(a_1 \cdots a_\ell) = NT(a_1) \cup \cdots \cup NT(a_\ell)$. For any $\eta \in \Sigma^*$, $CT(\eta) \stackrel{\text{def}}{=} (T_0 \cap ST(terms(\eta))) \setminus NT(\eta)$ is the set of *constants* of η. An *event* is a pair (η, i) where $\eta \in \Sigma^+$ and $1 \leq i \leq |\eta|$. The set of all events is called $Events$. For $e = (a_1 \cdots a_\ell, i) \in Events$, $act(e) = a_i$.

Note that B is (merely) the intended receiver in $A!B{:}(M)t$ and the purported sender in $A?B{:}t$. As we will see later, every send action is an instantaneous receive by the intruder, and similarly, every receive action is an instantaneous send by the intruder.

Σ_A, the set of A-*actions* is given by $\{C!D{:}(M)t, C?D{:}t \in \Sigma \mid C = A\}$. For any $\eta = a_1 \cdots a_\ell \in \Sigma^*$ and any $A \in Ag$, $\eta \upharpoonright A$ is given by $a_{i_1} \cdots a_{i_r}$ where $\{i_1, \ldots, i_r\} = \{i \leq \ell \mid a_i \in \Sigma_A\}$.

Definition 2.1 *Let $T \subseteq \mathcal{T}$ be a set of information terms.* $\mathsf{analz}(T)$*, the set of terms* analyzable *from T, is the least subset $\widehat{T}$ of $\mathcal{T}$ such that: $T \subseteq \widehat{T}$; if $(t_1, t_2) \in \widehat{T}$ then $t_1 \in \widehat{T}$ and $t_2 \in \widehat{T}$; and if $\{t\}_k \in \widehat{T}$ and $\overline{k} \in \widehat{T}$ then $t \in \widehat{T}$.*

$\mathsf{synth}(T)$*, the set of terms* synthesizable *from T, is the least subset $\widehat{T}$ of $\mathcal{T}$ such that: $T \subseteq \widehat{T}$; if $t_1 \in \widehat{T}$ and $t_2 \in \widehat{T}$ then $(t_1, t_2) \in \widehat{T}$; and if $t \in \widehat{T}$ and $k \in \widehat{T}$ then $\{t\}_k \in \widehat{T}$. For ease of notation,* $\mathsf{synth}(\mathsf{analz}(T))$ *is denoted $\overline{T}$.*

The definitions of analz and synth are due to [Pau98]. We will assume a number of basic properties of synth and analz proved in [Pau98].

Definition 2.2 *An* information state *s is a tuple $(s_A)_{A \in Ag}$ where for each agent A, $s_A \subseteq \mathcal{T}$. $\mathcal{S}$ denotes the set of all information states. The notions of an action enabled at a state and update of a state on an action are given as follows:*

- *$A!B{:}(M)t$ is* enabled *at s iff $t \in \overline{s_A \cup M}$, and if none of the terms in M occurs in s.*
- *$A?B{:}t$ is* enabled *at s iff $t \in \overline{s_I}$.*
- *$update(s, A!B{:}(M)t) = s'$ where $s'_A = s_A \cup M$, $s'_I = s_I \cup \{t\}$, and $s'_C = s_C$ for all the other $C \in Ag$.*
- *$update(s, A?B{:}t) = s'$ where $s'_A = s_A \cup \{t\}$ and $s'_C = s_C$ for all other $C \in Ag$.*

We extend the notion of update to sequences of actions as follows: $update(s, \varepsilon) = s$, $update(s, \eta \cdot a) = update(update(s, \eta), a)$.

Definition 2.3 *A* **protocol** Pr *is a sequence $a_1 b_1 \cdots a_\ell b_\ell \in \Sigma^+$ such that:*

- *for all $i : 1 \leq i \leq \ell$, b_i is a_i's matching receive,*
- *for all $k \in K_{st}$ referred to in $ST(terms(\mathsf{Pr}))$, $k \in NT(\mathsf{Pr})$, and*
- *for $s_0 = (K_A \cup CT(\mathsf{Pr}))_{A \in Ag}$, for all $i : 1 \leq i \leq \ell$, a_i is enabled at $update(s_0, a_1 b_1 \cdots a_{i-1} b_{i-1})$.*

One of the standard presentations of protocols is as a sequence of *communications* of the form $A \rightarrow B : (M)t$. For technical convenience, we split each communication of the above form into a pair of actions, $A!B{:}(M)t$ and $B?A{:}t$. We also require that all the short-term keys used in the protocol are freshly generated. This is a standard requirement and explains precisely why these keys are called "short-term".

Given a protocol Pr, $Roles(\mathsf{Pr}) \overset{\text{def}}{=} \{\mathsf{Pr} \upharpoonright A \mid A \in Ag \text{ and } \mathsf{Pr} \upharpoonright A \neq \varepsilon\}$.

A *substitution* σ is a map from T_0 to $\mathcal{T}$ such that: $\sigma(Ag) \subseteq Ag$, if $A \neq B$ then $\sigma(A) \neq \sigma(B)$, $\sigma(K_{st}) \subseteq K_{st}$, $\sigma(k_{AB}) = k_{\sigma(A)\sigma(B)}$, $\sigma(pubk_A) = pubk_{\sigma(A)}$, and $\sigma(privk_A) = privk_{\sigma(A)}$. Substitutions are extended to terms and actions pointwise. σ is *suitable* for a iff for all $m \in NT(a)$, $\sigma(m) \in N$ and for $m \neq n \in NT(a)$, $\sigma(m) \neq \sigma(n)$. For $\eta = a_1 \cdots a_\ell \in \Sigma^*$, σ is suitable for η iff it is suitable for a_i for all $i \leq \ell$, and $\sigma(\eta) = \sigma(a_1) \cdots \sigma(a_\ell)$. A substitution σ is said to be *suitable for* a protocol Pr if for all $t \in CT(\mathsf{Pr}), \sigma(t) = t$.

The important point here is that if an action a' with $term(a') = t'$ is an instance of an action a with $term(a) = t$, then t' has the same "structure" as t (i.e., t' is a substitution instance of t), but t' might be longer than t. This arises because the intruder substitutes a longer term in place of a nonce in t. In such a situation the behaviour of the honest agents is as if t' is of the same length as t – since the honest agents follow the protocol and hence do not expect any message communicated to have a longer term in place of a nonce. Note that the definition of substitutions being suitable for an action ensures that honest agents only substitute nonces for nonces, acting according to protocol.

Given a protocol Pr, $\eta' \in \Sigma^*$ is a *play* of Pr if $\eta' = \sigma(\eta)$ where $\eta \in \mathit{Roles}(\mathsf{Pr})$ and σ is a substitution suitable for Pr and η. $\mathit{Plays}(\mathsf{Pr})$ is the set of all plays of Pr. $\mathit{Events}(\mathsf{Pr}) = \{(\eta, i) \in \mathit{Events} \mid \eta \in \mathit{Plays}(\mathsf{Pr})\}$.

Define a function *infstate* from $\mathcal{S} \times \mathit{Events}(\mathsf{Pr})^*$ to $\mathcal{S}$ by induction as follows:

- $\mathit{infstate}(s_0, \varepsilon) = s_0$.
- If $\mathit{infstate}(s_0, \xi) = s$ and $\xi' = \xi \cdot e$, then $\mathit{infstate}(s_0, \xi') = \mathit{update}(s, \mathit{act}(e))$.

If $\mathit{infstate}(s_0, \xi) = s$, for any $A \in \mathit{Ag}$, $\mathit{infstate}_A(s_0, \xi) = s_A$.

Given a protocol Pr, $s_0 \in \mathcal{S}$ is said to be an *initial information state* of Pr if for all $A \in \mathit{Ho}$, $(s_0)_A = K_A \cup \mathit{CT}(\mathsf{Pr})$ and there exists a subset T of $N \cup K_{st}$ such that $(s_0)_I = K_I \cup \mathit{CT}(\mathsf{Pr}) \cup T$. The set of all initial information states of Pr is denoted by $\mathsf{Init}(\mathsf{Pr})$.

Definition 2.4 *Given a protocol* Pr*, the set of* runs *of* Pr*,* $\mathcal{R}(\mathsf{Pr})$ *is inductively defined as follows:*

- $(s_0, \varepsilon) \in \mathcal{R}(\mathsf{Pr})$ *for every* $s_0 \in \mathsf{Init}(\mathsf{Pr})$.
- *Suppose* $(s_0, \xi) \in \mathcal{R}(\mathsf{Pr})$ *and* $\mathit{infstate}(s_0, \xi) = s$. *Suppose there is* (η, i) *such that for all* $1 \leq j < i$, (η, j) *occurs in* ξ, (η, i) *does not occur in* ξ, *and* $\mathit{act}(\eta, i)$ *is enabled at* s. *Then* $(s_0,\ \xi \cdot (\eta, i)) \in \mathcal{R}(\mathsf{Pr})$.

Definition 2.5 *Given a protocol* Pr, $\mathsf{Sys}(\mathsf{Pr}) = (Q, I, \longrightarrow)$ *is the* system *defined by it, where:*

- Q*, the set of* protocol states*, is* $\mathcal{R}(\mathsf{Pr})$.
- I*, the set of* initial protocol states*, is* $\{(s, \varepsilon) \mid s \in \mathsf{Init}(\mathsf{Pr})\}$.
- *for* $(s, \xi), (s', \xi') \in Q$ *and* $a \in \Sigma$, $(s, \xi) \xrightarrow{a} (s', \xi')$ *iff* $s = s'$ *and there exists* (η, i) *such that* $\xi' = \xi \cdot (\eta, i)$ *and* $\eta(i) = a$.

Definition 2.6 *Suppose* Pr *is a protocol and let* $\mathsf{Sys}(\mathsf{Pr})$ *be* $(Q, I, \longrightarrow)$. *For* $q \in Q$ *and* $m \in T_0$, *we say that* m *is a* secret *at* q *if there exists* $A \in \mathit{Ho}$ *such that* m *belongs to* $\mathsf{analz}(s_A) \setminus \mathsf{analz}(s_I)$ *(where* $\mathit{infstate}(q) = s$*).* Pr *is* leaky *iff there exist* $q, q' \in Q$ *with* $q \xrightarrow{*} q'$ *and* $m \in T_0$ *such that* m *is a secret at* q *and* $m \in \overline{\mathit{infstate}_I(q')}$. Pr *is said to* preserve secrecy *iff it is non-leaky.*

The **secrecy problem** is the problem of determining whether a given protocol preserves secrecy.

3 An equivalence on terms

In this section we define an equivalence relation on terms and prove some of its properties.

Say that *a key k encrypts in a term t* if $\exists t' : \{t'\}_k \in \mathit{ST}(t)$. Given a term t and a key k define t_{-k} by induction as follows: for $m \in T_0$, $m_{-k} = m$; $(t, t')_{-k} = (t_{-k}, t'_{-k})$; and $(\{t\}_{k'})_{-k}$ is defined to be t_{-k} if $k = k'$, and $\{t_{-k}\}_{k'}$ otherwise. The binary relation $\equiv$ on terms is given in Figure 1. We say that $t \equiv_1 t'$ iff there is a "proof" of $t \equiv t'$ which does not use the axioms (A2) or (A5). Any term which has a subterm of the form (t, t) or of the form $\{t\}_k$ with k encrypting in t is said to be a *redex*.

$$\begin{array}{lll}
\text{(A1) } t \equiv t & \text{(R1) } \dfrac{t \equiv t'}{t' \equiv t} & \text{(R3) } \dfrac{t_1 \equiv t_1', \quad t_2 \equiv t_2'}{(t_1, t_2) \equiv (t_1', t_2')} \\
\text{(A2) } (t, t) \equiv t & & \\
\text{(A3) } (t, t') \equiv (t', t) & & \\
\text{(A4) } (t, (t', t'')) \equiv ((t, t'), t'') & \text{(R2) } \dfrac{t \equiv t', \quad t' \equiv t''}{t \equiv t''} & \text{(R4) } \dfrac{t \equiv t'}{\{t\}_k \equiv \{t'\}_k} \\
\text{(A5) } \{t\}_k \equiv \{t_{-k}\}_k & &
\end{array}$$

Fig. 1. Definition of $\equiv$.

Definition 3.1 *A term t is said to be* normal *if there is no t' such that $t \equiv_1 t'$ and t' is a redex. An action a is normal iff $term(a)$ is normal. A sequence of actions $a_1 \ldots a_\ell$ is normal iff for all $i \leq \ell$, a_i is normal.* $\mathsf{Pr} = a_1 b_1 \cdots a_\ell b_\ell$ *is called a* **normal protocol** *if* Pr *is a protocol and is also normal. An event (η, i) is normal if η is normal. A sequence of events $\xi = e_1 \cdots e_\ell$ is normal if for all $i \leq \ell$, e_i is normal.*

The main function of the equivalence relation is to ensure two things: the tupling operator works with sets of terms now rather than lists, which is ensured by Axioms (A2) to (A4); the depth of the encryption operator is bounded. The justification for the latter stems from the fact that honest agents, when they send terms on their own, would substitute only nonces for nonces, and hence would generate only terms of bounded encryption depth. The intruder would substitute terms of arbitrary depth for small terms, but this is precisely where, for purposes of detecting leaks, we can work with bounded depth terms, as the subsequent development will show.

Proposition 3.2 *The equivalence relation $\equiv$ on terms is of finite index.*

Proof Idea: It is easy to see that every term is equivalent to a normal term. We now show that the set of normal terms is finite, which will immediately imply the statement of the proposition.

Let $|T_0| = B$. Let N_i denote the set of normal terms of encryption depth i. It suffices to show that N_B is finite, since no term t of encryption depth greater than B can be normal. N_0 is just the set of tuples of distinct terms from of T_0, and hence $|N_0| \leq 2^{O(B)}$. Now for any i, N_{i+1} is got by encrypting terms from N_i using at most B keys and forming tuples of such terms which are distinct. We can show that the number of terms that can be formed in this manner is at most $2^{O((B+1) \cdot |N_i|)}$. Thus by induction, N_i is a finite set for all $i \geq 0$. Hence the result.

□

The above proof also tells us that the size of any normal term is bounded. Let us denote that bound by M for the rest of the paper. (M depends only on the size of T_0, which is fixed for the discussion in the paper.)

If $a = A!B{:}(M)t$ and $a' = A'!B'{:}(M')t'$ then $a \equiv a'$ iff $A = A', B = B', M = M'$ and $t \equiv t'$. Similarly for $a = A?B{:}t$ and $a' = A'?B'{:}t'$, $a \equiv a'$ iff $A = A', B =$

B' and $t \equiv t'$. Now for $\eta = a_1 \cdots a_\ell$ and $\eta' = a'_1 \cdots a'_\ell$, $\eta \equiv \eta'$ iff for all $i \leq \ell$, $a_i \equiv a'_i$. For any two $(\eta, i), (\eta', i) \in \mathit{Events}$, we say that $(\eta, i) \equiv (\eta', i')$ iff $\eta \equiv \eta'$ and $i = i'$. We extend $\equiv$ to Events^* as follows: $\equiv$ is the least equivalence relation on Events^* such that

- for $\xi = e_1 \cdots e_\ell$ and $\xi' = e'_1 \cdots e'_\ell$, if for all $i \leq \ell$, $e_i \equiv e'_i$ then $\xi \equiv \xi'$.
- $\xi \cdot e \equiv \xi$ if e occurs in ξ.
- if $\xi \equiv \xi'$ then for any e, $\xi \cdot e \equiv \xi' \cdot e$.

$(s_0, \xi) \equiv (s'_0, \xi')$ iff $s_0 = s'_0$ and $\xi \equiv \xi'$.

Proposition 3.3 *Suppose* Pr *is a protocol and* $(s_0, \xi), (s'_0, \xi') \in \mathcal{R}(\mathsf{Pr})$ *such that* $(s_0, \xi) \equiv (s'_0, \xi')$. *Then for any* $m \in T_0$ *and* $A \in Ag$, $m \in \mathit{infstate}_A(s_0, \xi)$ *iff* $m \in \mathit{infstate}_A(s'_0, \xi')$.

Proof Idea: Throughout the proof, we use the fact that since $(s_0, \xi) \equiv (s'_0, \xi')$, $s_0 = s'_0$. It is trivial to see that if e occurs in ξ' and $\xi = \xi' \cdot e$, then $\mathit{infstate}(s_0, \xi) = \mathit{infstate}_A(s'_0, \xi')$. It suffices to prove the proposition in the case when $\xi = e_1 \cdots e_\ell$, $\xi' = e'_1 \cdots e'_\ell$ and for all $i \leq \ell$, $e_i \equiv e'_i$. We can then argue by induction on the "proof" that $(s_0, \xi) \equiv (s'_0, \xi')$.

Suppose $\xi = e_1 \cdots e_\ell$, $\xi' = e'_1 \cdots e'_\ell$ and for all $i \leq \ell$, $e_i \equiv e'_i$. We now proceed by induction on ℓ. In the base case $\xi = \xi' = \varepsilon$ and therefore clearly $\mathit{infstate}(s_0, \xi) = \mathit{infstate}(s_0, \xi') = s_0$. For the induction step suppose that $\xi = \xi_1 \cdot e$ and $\xi' = \xi'_1 \cdot e'$ with $e \equiv e'$ and that for all $A \in Ag$ and $m \in T_0$, $m \in \mathsf{analz}(s_A)$ iff $m \in \mathsf{analz}(s'_A)$ (letting $\mathit{infstate}(s_0, \xi) = s$ and $\mathit{infstate}(s_0, \xi') = s'$). Consider the case when $e = (\eta, i)$ with $\eta(i) = A?B{:}t$ and $e' = (\eta', i)$ with $\eta'(i) = A?B{:}t'$ and $t \equiv t'$. Now we have to prove that for all $m \in T_0$, $m \in \mathsf{analz}(s_A \cup \{t\})$ iff $m \in \mathsf{analz}(s'_A \cup \{t'\})$. This is easily seen to be true using the fact that if t and t' are equivalent terms, then the same set of basic terms occur as subterms in both t and t' and further every such basic term m is encrypted by the same set of keys in both t and t', as is evident from the axiom (A5) in the definition of $\equiv$. The case when $\eta(i)$ and $\eta'(i)$ are send actions is treated in identical fashion.

□

Note that the fact that $\equiv$ is of finite index crucially depends on the fact that T_0 is finite. When we consider the situation where N and K_{st} are infinite, Proposition 3.2 does not hold. Nevertheless, we can build an equivalence of finite index on terms starting from a given equivalence relation on T_0. Thus the above results can be adapted to a more general setting also.

4 Decidability

Let Pr be a protocol and let $\mathsf{Sys}(\mathsf{Pr}) = (Q, I, \longrightarrow)$. The set $\mathcal{R}'(\mathsf{Pr})$ is defined to be $\{(s_0, \xi) \mid \xi$ is normal and there exists ξ' such that $\xi \equiv \xi'$ and $(s_0, \xi') \in \mathcal{R}(\mathsf{Pr})\}$. The finite state system $\mathsf{Sys}'(\mathsf{Pr})$ as defined to be $(\mathcal{R}'(\mathsf{Pr}), I, \longrightarrow')$ where for $(s, \xi), (s', \xi') \in \mathcal{R}'(\mathsf{Pr})$ and $a \in \Sigma$, $(s, \xi) \xrightarrow{a}{}' (s', \xi')$ iff $s = s'$ and there exists

(η, i) such that $\xi' = \xi \cdot (\eta, i)$ and $\eta(i) = a$. We say that $\mathsf{Sys}'(\mathsf{Pr})$ is leaky iff there are two states $r, r' \in \mathcal{R}'(\mathsf{Pr})$ with $r \xrightarrow{*}{}' r'$ and $m \in T_0$ such that m is a secret of r and $m \in \mathsf{analz}(\mathit{infstate}_I(r'))$.

The following lemma is an easy consequence of Proposition 3.2.

Lemma 4.1 $\mathsf{Sys}'(\mathsf{Pr})$ *is a finite-state system.*

Theorem 4.2 Pr *is leaky iff* $\mathsf{Sys}'(\mathsf{Pr})$ *is leaky.*

Proof Idea: Suppose $\mathsf{Sys}(\mathsf{Pr}) = (Q, I, \longrightarrow)$ and $\mathsf{Sys}'(\mathsf{Pr}) = (Q', I', \longrightarrow')$. Suppose Pr is leaky. This means that there are two states $q, q' \in Q$ with $q \xrightarrow{*} q'$ and $m \in T_0$ such that m is a secret of q and $m \in \mathsf{analz}(\mathit{infstate}_I(q'))$. Therefore there is a state $r' \in Q'$, such that $q' \equiv r'$. Now since $q \xrightarrow{*} q'$, there exist $s_0 \in \mathcal{S}$ and ξ, ξ' such that $q = (s_0, \xi)$, $q' = (s_0, \xi')$ and ξ is a prefix of ξ'. Let $r' = (s_0, \varsigma')$. Since $q' \equiv r'$, $\xi' \equiv \varsigma'$. We can easily argue that there is a prefix ς of ς' such that $\xi \equiv \varsigma$. Let $r = (s_0, \varsigma)$. Then $q \equiv r$. Now it follows from Lemma 3.3 that m is a secret of r and $m \in \mathsf{analz}(\mathit{infstate}_I(r'))$. Thus $\mathsf{Sys}_{\equiv}(\mathsf{Pr})$ is leaky.

Suppose $\mathsf{Sys}'(\mathsf{Pr})$ is leaky. Then there are two $r, r' \in Q'$ with $r \xrightarrow{*}{}' r'$ and $m \in T_0$ such that m is a secret of r and $m \in \mathsf{analz}(\mathit{infstate}_I(r'))$. We know that there is some $q' \in Q$ such that $q' \equiv r'$. From Lemma 3.3, it follows that $m \in \mathsf{analz}(\mathit{infstate}_I(q'))$. Let $r = (s_0, \varsigma)$ and $r' = (s_0, \varsigma')$. Let $q' = (s_0, \xi')$. We can easily argue that there is a prefix ξ of ξ' with $\xi \equiv \varsigma$. Let $q = (s_0, \varsigma)$. Then $q \equiv r$ and therefore m is a secret of q as well, by Lemma 3.3. Thus Pr is leaky.

□

Lemma 4.3 *Let* Pr *be a normal protocol,* $(s_0, \xi) \in \mathcal{R}(\mathsf{Pr})$*, and* $M' = max_{t \in s}|t|$ *where* $\mathit{infstate}(s_0, \xi) = s$*. Then for every* e *enabled at* (s_0, ξ)*, there is* $e' \equiv e$ *such that* $|\mathit{term}(e')| \leq M \cdot (M' + 1)$ *and* e' *is enabled at* (s_0, ξ)*.*

Proof Idea: This is trivial to see if e corresponds to a send action. Since the protocol is normal, a send by an honest agent, which involves a normal term having size at most M and mentioning at most M nonces, is of size at most $M \cdot (M' + 1)$ since according to semantics the nonces in the term are substituted either with newly generated nonces or with terms already received.

Suppose e corresponds to a receive action. This corresponds to a send by the intruder. There is no a priori bound on the size of e. But since the protocol description mentions only normal terms, there is always some event $e' \equiv e$ enabled at (s_0, ξ) such that for some normal term $t \in \overline{s_I}$, $\mathit{term}(e')$ is got by substituting terms from $\mathsf{analz}(s_I)$ for nonces in t. It can be easily seen that $\mathit{term}(e')$ is of size at most $M \cdot (M' + 1)$.

□

Theorem 4.4 *For the class of normal protocols, the secrecy problem is decidable.*

Proof Idea: It suffices to prove that for the class of normal protocols, the set $\mathcal{R}'(\mathsf{Pr})$ is effectively constructible. We show by induction on i, how to construct the set of sequences of length i in $\mathcal{R}'(\mathsf{Pr})$. This suffices to prove that $\mathcal{R}'(\mathsf{Pr})$ is constructible, since all the sequences in it are of bounded length.

Let us denote by R_i the set of sequences in $\mathcal{R}'(\mathsf{Pr})$ of length i. R_0 is essentially the set of initial information states of Pr. This is easily constructible since each initial information state is a tuple of subsets of T_0, which is a fixed finite set.

Suppose R_i is constructible. Now we claim that R_{i+1} is exactly the set of $(s_0, \xi_1 \cdot e)$ such that $(s_0, \xi_1) \in R_i$, e is normal, and there exists e' with $|\mathit{term}(e')| \leq M \cdot (M+1)$, $e \equiv e'$ and e' is enabled at (s_0, ξ). This is easily seen to be an effectively checkable condition and hence if we prove the above claim the theorem is proved.

Suppose $(s_0, \xi_1 \cdot e) \in R_{i+1}$. This means that $(s_0, \xi_1) \in R_i$, i.e., there is some $(s_0, \xi_1') \in \mathcal{R}(\mathsf{Pr})$ such that $\xi_1 \equiv \xi_1'$. We can show that there is some e' enabled at (s_0, ξ_1') with $e \equiv e'$. Therefore, by Lemma 4.3 there is e'' such that $e' \equiv e''$, $|\mathit{term}(e'')| \leq M \cdot (M'+1)$ where $M' = \mathit{max}\{|t| \mid t \in \mathit{infstate}(s_0, \xi_1')\}$. Now the terms from $\mathit{infstate}(s_0, \xi_1')$ which are used to construct e'' have "normal counterparts" in $\mathit{infstate}(s_0, \xi_1)$. Therefore we can show that there is e''' equivalent to e'' which is of size $\leq M \cdot (M+1)$ and is enabled at (s_0, ξ_1) (the "normal counterpart" of (s_0, ξ_1')). Also $e \equiv e'''$. This proves one direction of the claim.

Suppose now that $(s_0, \xi_1) \in R_i$ and that there exists e' such that $|\mathit{term}(e')| \leq M \cdot (M+1)$, $e \equiv e'$ and e' is enabled at (s_0, ξ_1). Now there is some $(s_0, \xi_1') \in \mathcal{R}(\mathsf{Pr})$ with $(s_0, \xi_1') \equiv (s_0, \xi_1)$. We can show that all e_1 occurring in (s_0, ξ_1) are "normal counterparts" of events occurring in (s_0, ξ_1'). Hence we can show that there is some e'' such that $|\mathit{term}(e'')| \leq M \cdot (M'+1)$ where $M' = \mathit{max}\{|t| \mid t \in \mathit{infstate}(s_0, \xi_1')\}$, and $e' \equiv e''$ and e'' is enabled at (s_0, ξ_1'). Since $e \equiv e'$, $e \equiv e''$. Since $(s_0, \xi_1' \cdot e'') \in \mathcal{R}(\mathsf{Pr})$, $(s_0, \xi_1 \cdot e) \in R_{i+1}$.

This concludes the proof that for normal protocols, the secrecy problem is decidable.

□

Note that in the setting we are considering in this paper, using the equivalence on terms of finite index, given any system we can quotient it to obtain an equivalent finite state system. As observed earlier, in the case of unboundedly many nonces, the equivalence on terms would not be on finite index, and hence the quotiented systems would in general not be finite. In [RS03], we consider the secrecy problem in the situation where message length is bounded but the nonce set is infinite, and show that, for a suitable subclass of protocols – we call them structured protocols – even though they have infinitely many runs in general, still they have the property that if the system is leaky, then there is run of bounded length witnessing a leak. This suffices to prove decidabilty.

5 Undecidability

We show in this section that the secrecy problem is undecidable even when the set of nonces used in runs of the protocols is finite. This is shown by sketching

how a two-counter machine can be coded up using the protocol formalism. A two-counter machine is a machine with a finite set of control states and two counters holding natural number values. The actions of the machine are state transitions based on the current state and the values of the counters. In addition, during each transition, the counters may be incremented or decremented, and also tested for zero. We code up any given machine by a protocol whose roles typically look like the following:
$A?B{:}\{q, x, (z, y)\}_{k_{AB}}$; $\quad A!B{:}\{q', (z, x), y\}_{k_{AB}}$.

The above role represents a transition which changes state from q to q', increments the first counter and decrements the second. The intruder uses the fact that arbitrary terms can be substituted in place of nonces to mimic the behavior of the machines which use the output configuration of one transition as the input configuration of another. This is achieved by the intruder blocking the sends in some play of a role of the above kind, and forwarding it to a receive in some other play (of possibly the same role). This is the key to undecidability.

The terms q and q' in the above merit special mention. They are meant to encode states of the two-counter machine. Now since an arbitrary two-counter machine does not have a bound on the number of its states, and since we are working with a fixed set of nonces, we cannot code states directly. The q's have to be thought of as some complex terms. Then it is easy to see that even if we manage to encode the pair of counters using normal terms in the protocol description, there is a large enough two-counter machine (say, one with much larger than 2^{M^2} states), which cannot be faithfully coded up by any normal protocol, since there are not enough normal terms to code up even the set of states of the given machine. In conclusion, for every fixed two-counter machine, there is a T_0 and a normal protocol which uses terms built from T_0 which codes the machine, but if we fix T_0, then there are machines (those whose number of states is much more than the number of normal terms based on T_0) which cannot be coded by any normal protocol which uses terms built from T_0.

There is also a coding which uses unbounded encryption depth. This shows that the restriction which the equivalence relation $\equiv$ places on encryption is also essential for decidability.

6 A modal logic

We can show that the decidability result of the secrecy problem for normal protocols also extends to the verification problem for a simple modal logic, in which one can state security properties.

The formulas of the logic are given by:

$$\Phi ::= A \ \mathsf{has}\ t \mid \neg\alpha \mid \alpha \vee \beta \mid \langle a \rangle \alpha \mid \Diamond \alpha$$

where $A \in Ag$, $t \in \mathcal{T}$, $a \in \Sigma$.

The logic is interpreted on models which are of the form (Sys, τ) where: $\mathsf{Sys} = (Q, I, \longrightarrow)$ is a Σ-labelled transition system and τ is a valuation function from Q to $\mathcal{S}$.

Given a model $M = ((Q, I, \longrightarrow), \tau)$, a state $q \in Q$, and a formula α, the notion $(M, q) \models \alpha$ is defined inductively as usual: $(M, q) \models A\ \mathsf{has}\ t$ iff $t \in \mathsf{synth}(\mathsf{analz}((\tau(q))_A))$; $(M, q) \models \neg\alpha$ iff $(M, q) \not\models \alpha$; $(M, q) \models \alpha \vee \beta$ iff $(M, q) \models \alpha$ or $(M, q) \models \beta$; $(M, q) \models \langle a \rangle \alpha$ iff $\exists q'$ such that $q \xrightarrow{a} q'$ and $(M, q') \models \alpha$; $(M, q) \models \Diamond\alpha$ iff $\exists q'$ such that $q \xrightarrow{*} q'$ and $(M, q') \models \alpha$ (where $\xrightarrow{*}$ is the reflexive transitive closure of the relation got by taking the union of the relations $\xrightarrow{a}$).

For $M = ((Q, I, \longrightarrow), \tau)$, we say that $M \models \alpha$ if $(M, q) \models \alpha$ for all $q \in I$ For a protocol Pr and a formula α, we say that $\mathsf{Pr} \models \alpha$ iff $(\mathsf{Sys}(\mathsf{Pr}), \mathit{infstate}) \models \alpha$. The **verification problem** is the problem of determining whether $\mathsf{Pr} \models \alpha$, given a protocol Pr and a formula α.

Note that interesting security properties can be specified in the logic. For instance, the following formula says that a basic term m is a secret of some state of the model.

$$sec_m \overset{\text{def}}{=} \Diamond\Big(\bigvee_{A \in Ho} A\ \mathsf{has}\ m \quad \wedge \quad \neg(I\ \mathsf{has}\ m)\Big).$$

The following formula says that the given model is leaky.

$$leaky \overset{\text{def}}{=} \Diamond\Big(\bigvee_{m \in T_0} sec_m \quad \wedge \quad \Diamond(I\ \mathsf{has}\ m)\Big).$$

References

[ALV01] Amadio, R.M., Lugiez, D. and Vanackère, V., "On the symbolic reduction of processes with cryptographic functions", INRIA Research Report 4147, March 2001.

[Bol97] Bolignano, D., "Towards a mechanization of cryptographic protocol verification", *CAV'97*, LNCS 1254, 1997, 131-142.

[CC03] Comon, H. and Cortier, V., "Tree automata with one memory, set constraints, and cryptographic protocols", To appear in *Theoretical Computer Science*, 2003.

[CCM01] Comon, H., Cortier, V. and Mitchell, J.C., "Tree automata with One Memory, Set Constraints, and Ping-Pong Protocols", *ICALP 2001*, Crete, Greece, July 2001.

[CJ97] Clark, J. and Jacob, J., "A survey of authentication protocol literature", Web draft version 1.0, `http://www.cs.york.ac.uk./~jac`, 1997.

[CS02] Comon, H. and Shmatikov, V., "Is it possible to decide whether a cryptographic protocol is secure or not?", *Journal of Telecommunications and Information Technology*, 2002.

[DEK82] Dolev, D., Even, S. and Karp, R.M., "On the Security of Ping-Pong Protocols", *Information and Control*, vol 55, 57-68, 1982.

[DLMS99] Durgin, N.A., Lincoln, P.D., Mitchell, J.C. and Scedrov, A., "The undecidability of bounded security protocols", *Workshop on Formal Methods and Security Protocols (FMSP'99).* Electronic proceedings available at: `http://www.cs.bell-labs.com/who/nch/fmsp99/program.html`.

[EG83] Even, S. and Goldreich, O., "On the security of multi-party ping-pong protocols", Tech. Rep. 285, Technion - Israel Institute of Technology, 1983.

[Gou00] Goubault-Larrecq, J., "A method for automatic cryptographic protocol verification", *IPDPS*, LNCS 1800, 2000, 977-984.

[HT96] Heintze, N. and Tygar, J. D., "A model for secure protocols and their composition", *IEEE Trans. on Soft. Engg*, vol 22, 16-30, 1996.

[Low96] Lowe, G., "Breaking and fixing the Needham - Schroeder public key protocol using FDR", *TACAS 96*, LNCS 1055, 1996, 147-166.

[MCJ97] Marrero, W., Clarke, E. M. and Jha, S., "Model checking for security protocols", *DIMACS Workshop on design and verificaton of security protocols*, 1997.

[Me95] Meadows, C.A., "Formal Verification of Cryptographic Protocols: (A Survey)", *Asiacrypt '94*, LNCS 917, pp 133-150, 1995.

[Mo99] Monniaux, D., "Abstracting cryptographic protocols with tree automata", *Static analysis symposium*, LNCS 1694, 1999.

[MS01] Millen, J. and Shmatikov, V., "Constraint Solving for Bounded-Process Cryptographic Protocol Analysis", *CCS'01*, 2001, 166-175.

[NS78] Needham, R. and Schroeder, M., "Using Encryption for Authentication in Large Networks of Computers", *Communications of the ACM*, 21(12), 1978.

[Pau98] Paulson, L., "The inductive approach to verifying cryptographic protocols", *Journal of computer security*, vol 6, 1998, 85-128.

[RS03] Ramanujam, R. and Suresh, S.P., "A decidable subclass of unbounded security protocols", To appear in *Proceedings of WITS'03*, Warsaw, Poland, April, April 2003.

[RT01] Rusinowitch, M. and Turuani, M., "Protocol insecurity with finite number of sessions is NP-complete", INRIA Research Report 4134, March 2001.

[SC00] Syverson, P. F. and Cervesato, I., "The logic of authentication protocols", *FOSAD*, pp 63-106, 2000.